Build 1: Light Sensor

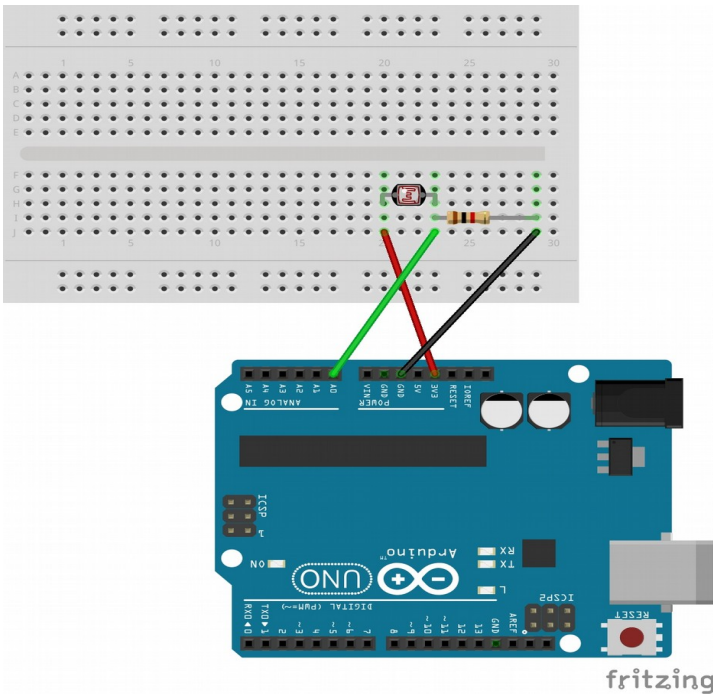
Description:

This was a light sensor build containing a photo cell light sensor, a 1KΩ resistor and three cords, one from the sensor to input A0, one from the resistor to ground, and one connecting the resistor and the sensor to the 3.3V point. The sensor would provide input to the serial monitor of the amount of light in the room every 5 seconds.

Issues:

There weren't really any issues with this build. I was able to get it up and running successfully with my first implementation. The only issue I ran into was that at some point, when I changed the resistor, all the readings became the same, around 1039. It wouldn't change regardless of the light level. I can't remember what resistor or change caused the issues because I was able to fix it within seconds of it starting. I was surprise that it worked so well.

Diagram of Build



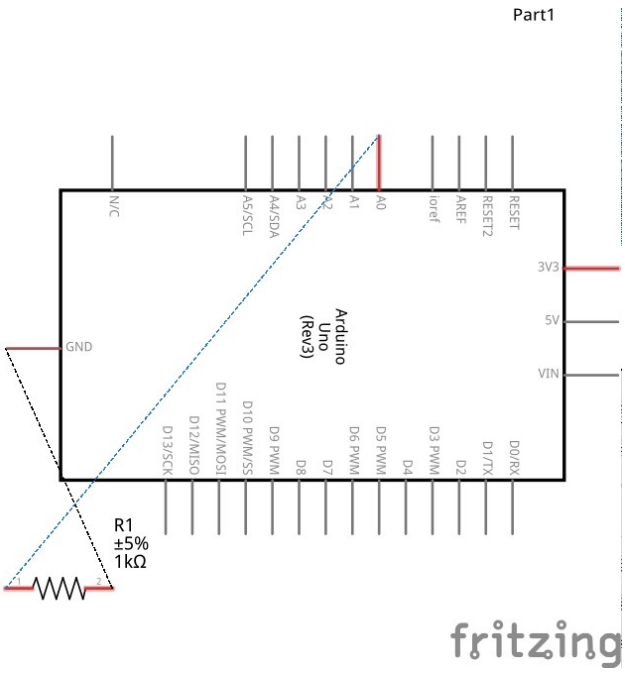
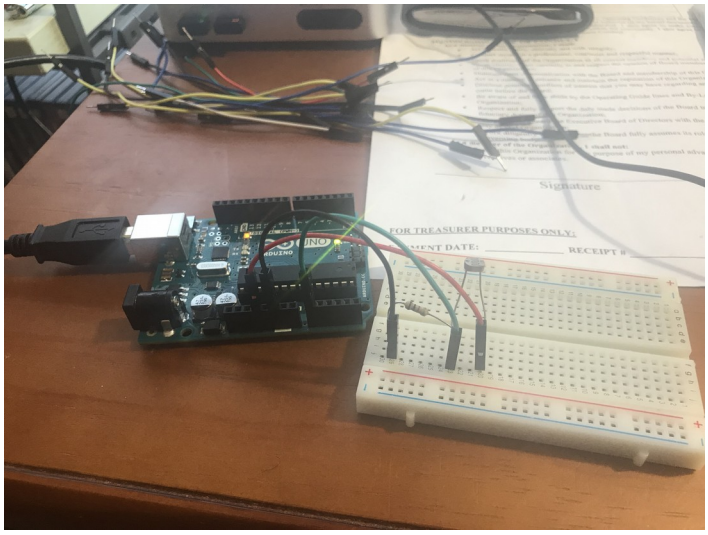


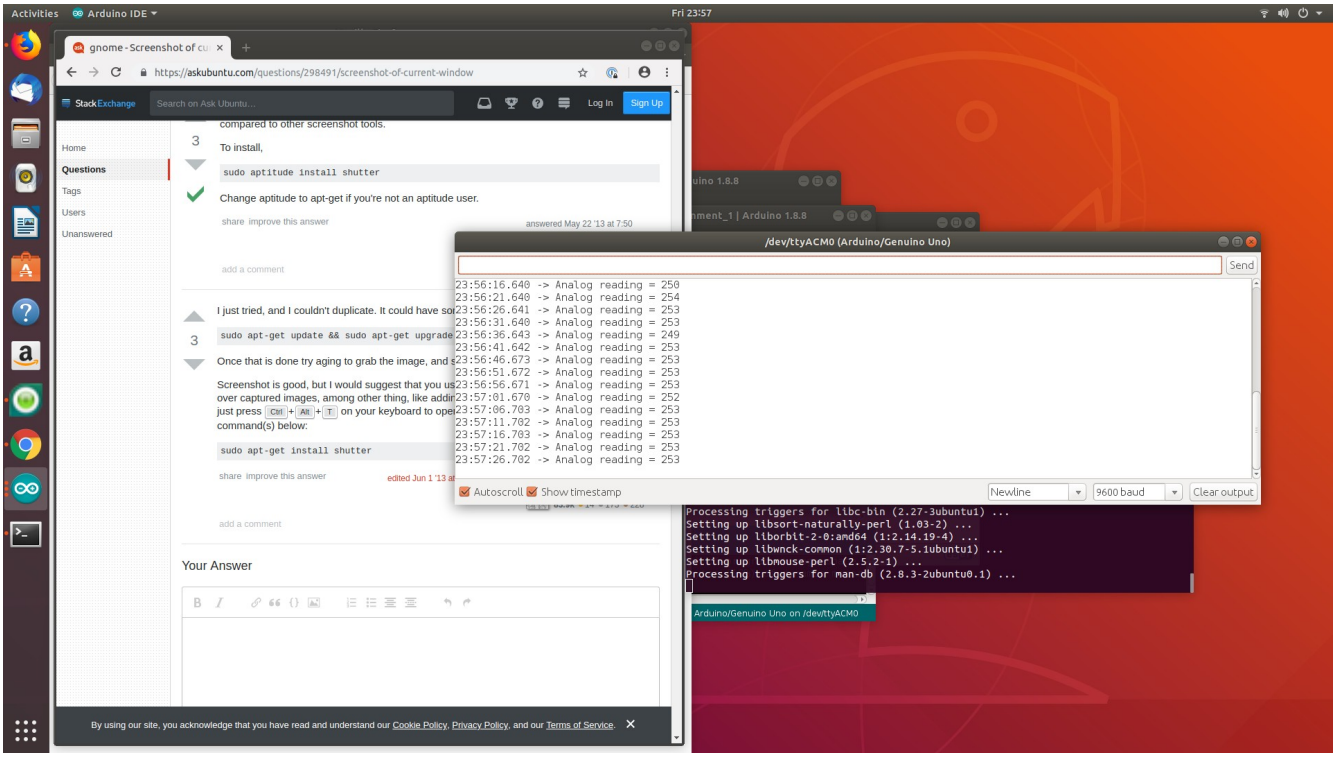
Photo of Build



Code for Build

```
/*        Most        of        this        code        is        from
https://learn.adafruit.com/photocells/arduino-code. I removed some of
the code pertaining to labeling the type of light,
changed line 15 to println instead of print and increased the delay
to 5000 instead of 1000.*/

int photocellPin = 0;      // the cell and 1K pulldown are connected
to a0
int photocellReading;        // the analog reading from the analog
resistor divider

void setup(void) {
  // We'll send debugging information via the Serial monitor
  Serial.begin(9600);
}

void loop(void) {
  photocellReading = analogRead(photocellPin);

  Serial.print("Analog reading = ");
  Serial.println(photocellReading);     // the raw analog reading

  delay(5000);
}
```

Example of Output



Build 3: LEDs

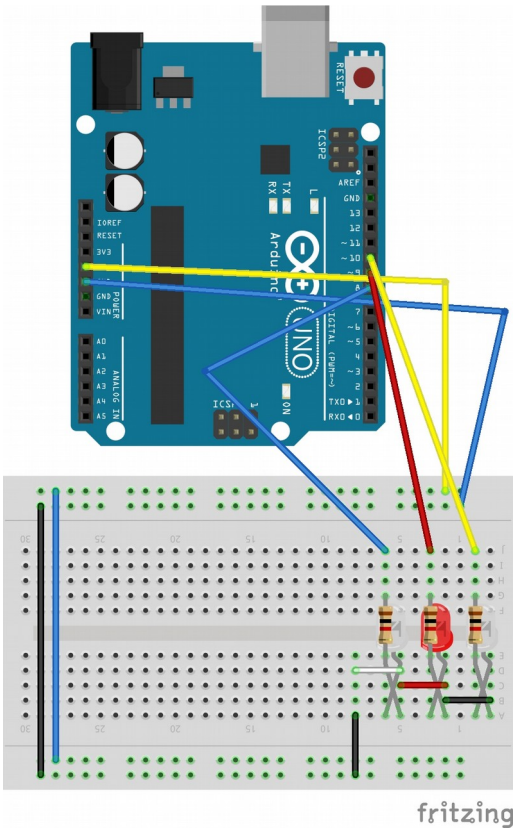
Description:

This was an LED build. The goal was to have three independent leds that would come on at different times. This build contained three leds, 9 wires and three 1KΩ resistors.

Issues:

So initially when I started working on the build, I had no idea where to start. So I found a resource that had an 8 led build using a shift register. I assumed I could adapt this to work with just three leds. Trying to put it together was so difficult because none of the resistors would touch and it took 30 minutes of my time until I found a simpler build and decided to use that instead. I was surprised at how easy it was once I found the simpler build. I was also surprised at how much I kept wanting to play with the lights once I got it working.

Diagram of Build



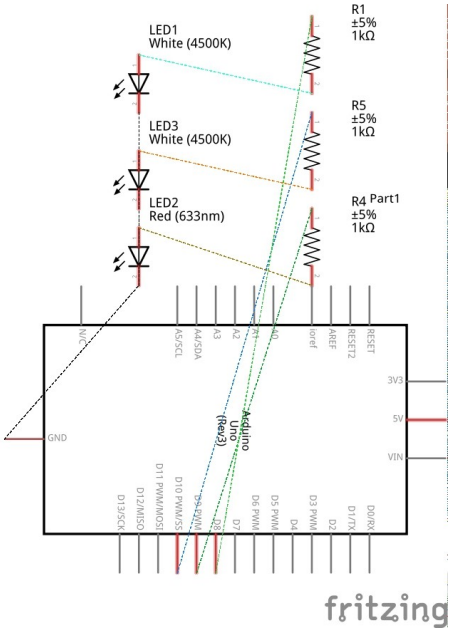
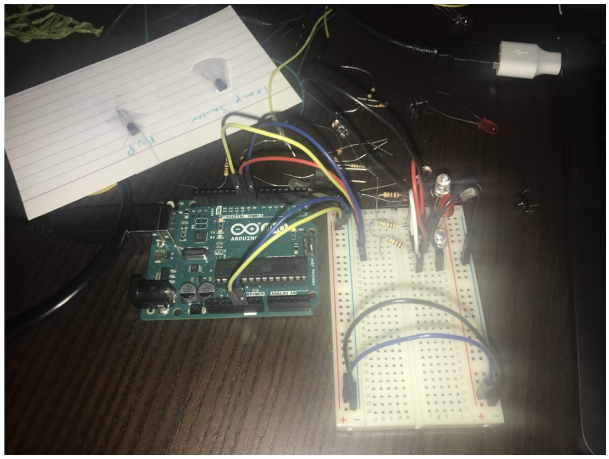


Photo of Build

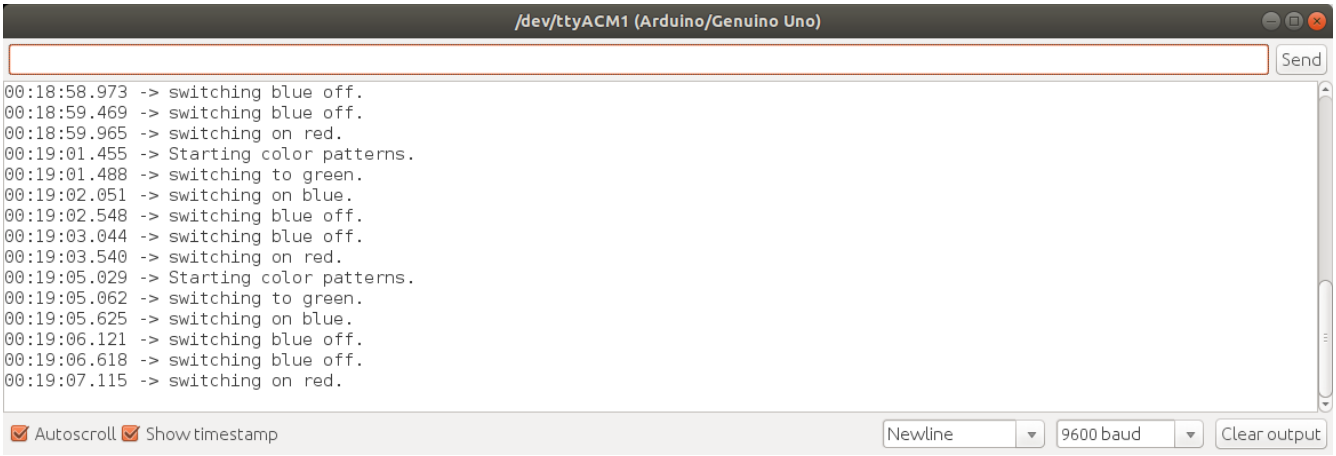


Code for Build

```
/*      The      basis      of      this      code      is      from
https://www.makeuseof.com/tag/arduino-traffic-light-controller/.
 *  I've edited the code so that it accounts for my different colored
leds and for my timing
 *  goals.*/

int blue = 8;
int red = 9;
int green = 10;

void setup() {
  Serial.begin(9600);
  pinMode(blue, OUTPUT);
  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);

}

void loop() {
  Serial.println("Starting color patterns.");
  changeLights();
  delay(1000);
}

void changeLights() {

  // turn off red, then turn on yellow for .5 seconds.
  Serial.println("switching to green.");
  digitalWrite(red, LOW);
  digitalWrite(green, HIGH);
  delay(570);
```

```
   //turn off yellow, then turn on blue twice, each time for .5
seconds, with a .5 second delay between them.
  Serial.println("switching on blue.");
  digitalWrite(green, LOW);
  digitalWrite(blue, HIGH);
  delay(500);
  Serial.println("switching blue off.");
  digitalWrite(blue, LOW);
  delay(500);
  Serial.println("switching blue off.");
  digitalWrite(blue, HIGH);
  delay(500);

  //turn off blue, turn on red for .5 seconds;
  Serial.println("switching on red.");
  digitalWrite(blue, LOW);
  digitalWrite(red, HIGH);
  delay(500);
}
```

Example of Output



Build 4: Motor

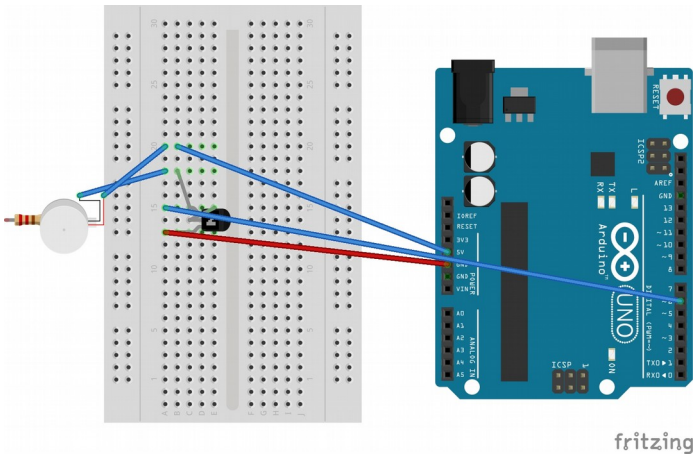
Description:

This was zamotor build where the goal was to get the motor vibration at intervals enough to shake an object. I accomplished this using an NPN, a mini motor, three cables, the arduino, and a resistor as the thing being shaken.

Issues:

The biggest issue I had was how painful it was working with the motor. When the motor was running at full strength, it was really hot, and hard to hold. It also made it difficult to check the circuit because it would shake so hard that it'd shake one or both of the wires connected to it out of the breadboard. I learned a lot about adjusting via analogWrite to get it to a point where it wasn't a danger to myself and anyone near me.

Diagram of Build



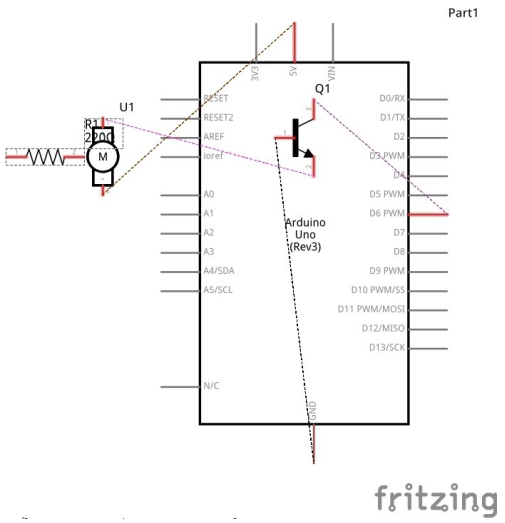
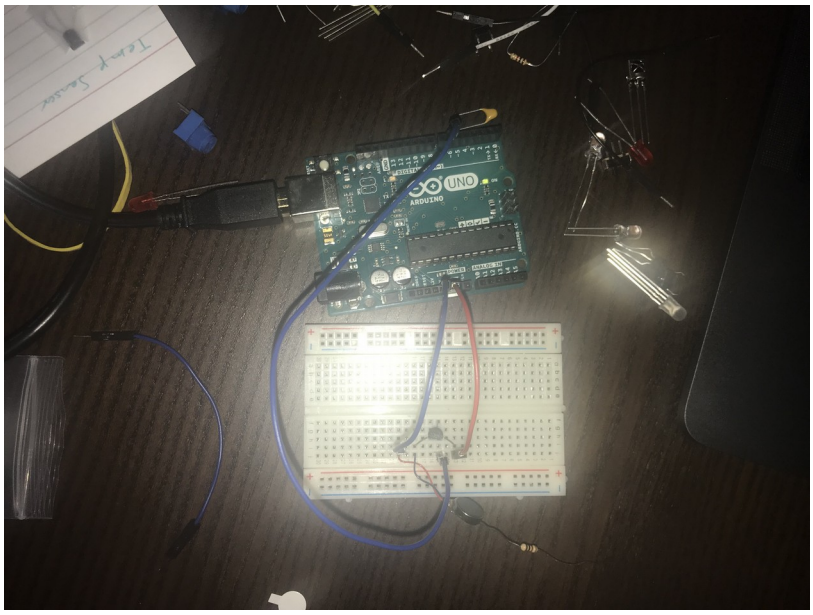


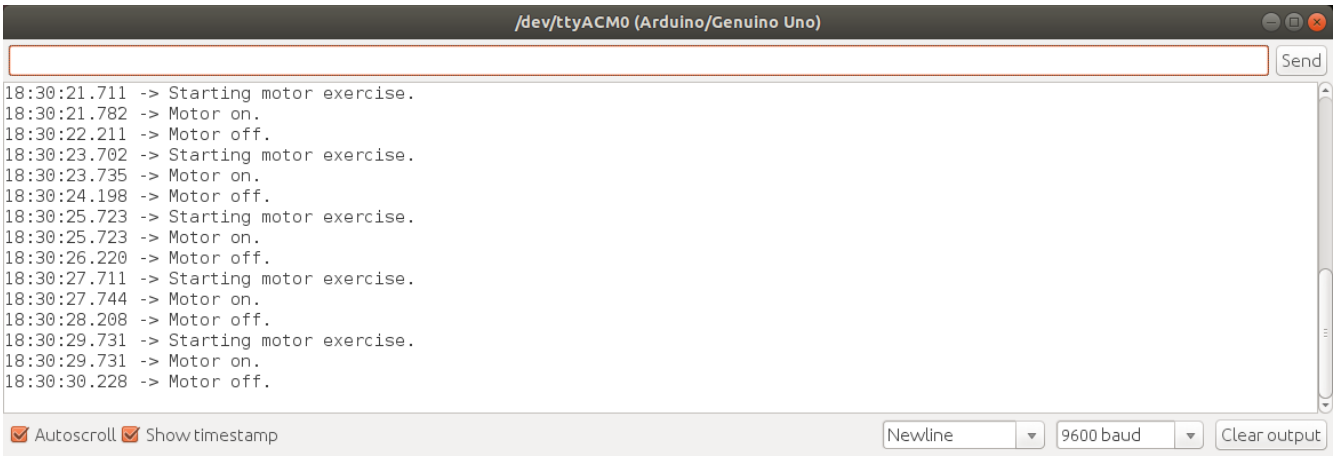
Photo of Build



Code for Build

```
/*     Developed     in     part     with     information     from
https://www.precisionmicrodrives.com/content/how-to-drive-a-
vibration-motor-with-arduino-and-genuino/.
I  didn't  copy  any  of  the  code,  just  used  analogWrite  as  they
suggested.*/

int motor = 6;

void setup() {
  Serial.begin(9600);
  pinMode(motor, OUTPUT);
}

void loop() {
 Serial.println("Starting motor exercise.");
 motorMoving();
 delay(1000);
}

void motorMoving() {
  //switch the motor on.
  Serial.println("Motor on.");
  analogWrite(motor, 50);
  delay(500);

  //switch the motor off.
  Serial.println("Motor off.");
  analogWrite(motor, 0);
  delay(500);
}
```

Example of Output



Build 5: Wildcard

Description:

For this project, I decided to see if I could dim the light out coming from an LED over time and bring it back up to its original brightness while keeping track of it as the intensity changes. The build was made with an LED, a 220Ω resister, the photosensor, a 10KΩ resistor, and 5 cables, one to ground one to digital (PWM~) 13, one to analog input A0, one to ground and one to 3.3 V.

Issues:

The biggest issue I had was trying to determine whether or not the light was dimming. Even though I had the serial monitor output showing how much power I was sending to the led each time, visually, I couldn't see whether it was dimming or not because the visual gradations seemed too minimal to see with the human eye. I ultimately decided to include the photosensor build from the first build to see if it could show the difference. I was surprised at how small the change in the power reading was using a 10KΩ resister. The fluctuations were all in single digits and it wasn't that much of a change between each power shift. I was also surprised by how many lights I had to shut off to get any type of reading from the photo sensor. If I didn't have all the lights off, I couldn't see any change.

Diagram of Build

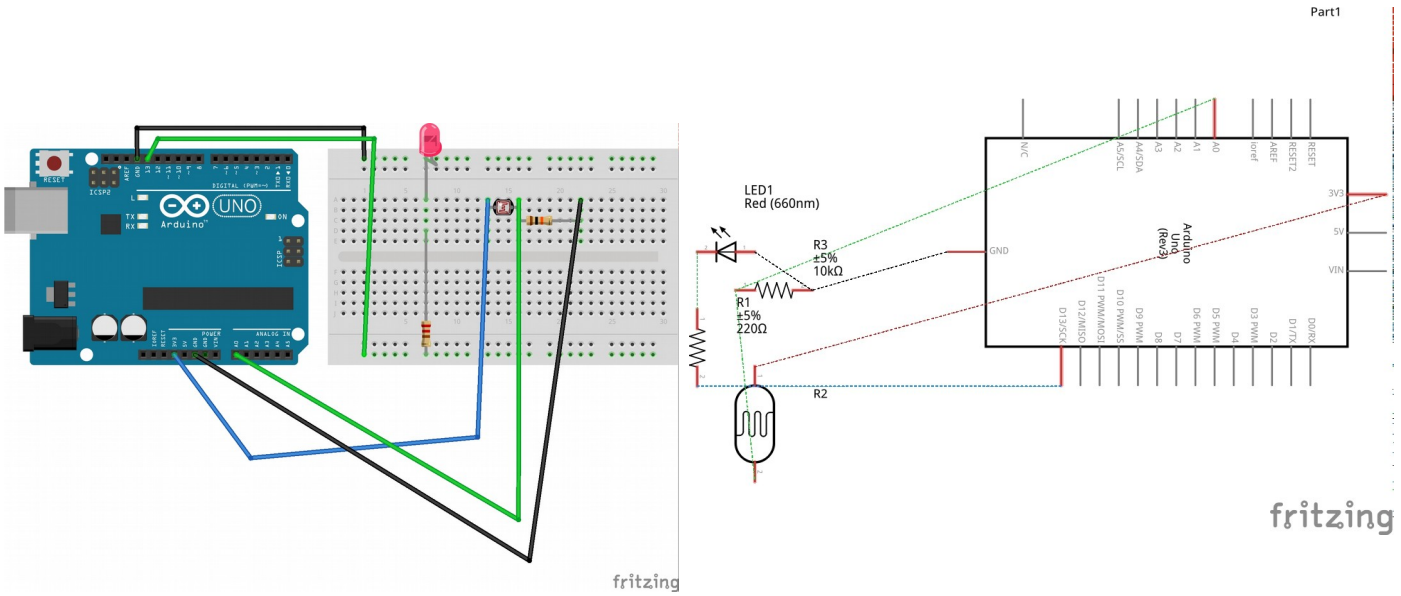
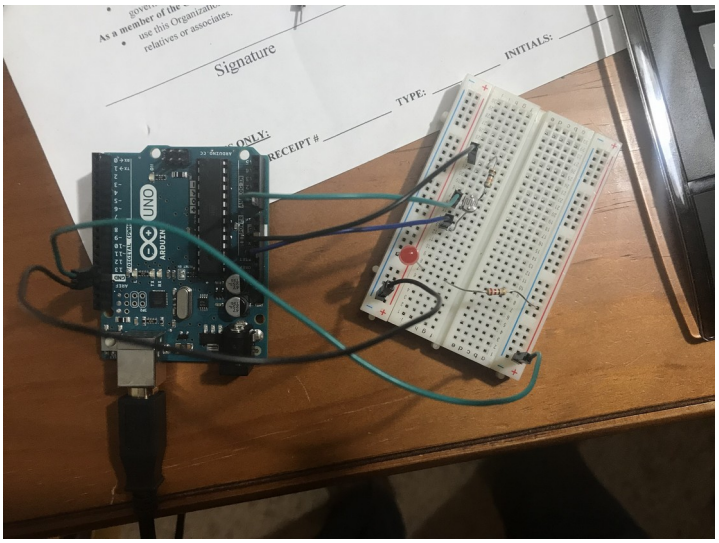


Photo of Build



Code for Build

```
/*
  Dims an led down and this raises the led back to full brightness.
            A      lot      of      the      code      is      from
https://www.arduino.cc/en/Tutorial/Fading.

  */

int led = 13;
int fadeInValue;
int fadeOutValue;
int photocellPin = 0;      // the cell and 1K pulldown are connected
to a0
int photocellReading;        // the analog reading from the analog
resistor divider

void setup() {
  // initialize the digital pin as an output.
  Serial.begin(9600);
  Serial.println("Fading exercise.");
  pinMode(led, OUTPUT);
  delay(3000);
}

void loop() {
  Serial.println("Fade in.");
  for (fadeInValue = 0; fadeInValue <= 255; fadeInValue += 10) {
    analogWrite(led, fadeInValue);

    photocellReading = analogRead(photocellPin);
    Serial.print("Analog reading = ");
    Serial.println(photocellReading);
```

```
    String statement = "Fade in value: ";
    String statement2 = statement + fadeInValue;
    Serial.println(statement2);
    delay(570);
  }

  Serial.println("Fade out.");
  for (fadeOutValue = 255; fadeOutValue >= 0; fadeOutValue -= 10) {
    analogWrite(led, fadeOutValue);

    photocellReading = analogRead(photocellPin);
    Serial.print("Analog reading = ");
    Serial.println(photocellReading);

    String statement3 = "Fade out value: ";
    String statement4 = statement3 + fadeOutValue;
    Serial.println(statement4);
    delay(570);
  }
}
```

Example of Output